CS 6453: Haven

Soumya Basu
April 25, 2017
Motivation

• Applications are being migrated to the cloud

• People want to outsource sensitive applications there as well
  • User data, Medical Records, Private Keys, etc.

• Need to trust the cloud provider to keep things hidden and secure…
  • Not always possible (e.g. Aurora attacks)
New Technology

• Intel SGX- New Secure Instructions on Intel Chip
Hardware Isolation

- Hardware Security Modules offer limited functionality
- Trusted Platform Modules have two drawbacks
  - Vulnerable to relatively easy hardware attacks
  - Bad multiplexing between secure/insecure environments
- Two approaches to multiplexing:
  - Time-multiplex the entire page cache (inefficient!)
  - Attest to the OS code that is running (prevents upgradability)
Software Isolation

• General approach: protect apps against untrusted OS

• Systems that use this approach are vulnerable to Iago attacks
  • Only defense: write a shield module against OS

• Haven does something similar, but leverages SGX
Untrusted Cloud

• Can use crypto techniques to outsource work to cloud
  • Only works for new applications
  • Very hard to reason about security
Where is this work new?

• Define the concept of shielded execution, see how SGX plays into this picture

• Unmodified applications, which means OS functionality has to be untrusted
  • Private scheduling of threads
  • Virtual Memory Management untrusted
  • Encrypted and Integrity-protected file system
Shielded Execution

- Confidentiality
  - Execution of a program is a black box
  - Inputs and outputs observable, no intermediate state
  - Why is this good enough?

- Integrity: System cannot change program behavior
- What about availability?
Threat Model

- Malicious cloud
  - Controls all code outside of client app code
    - Software stack, Host OS, scheduling, etc.
  - Intel is trustworthy, secure hardware is secure
- Side channel attacks not covered
• Memory Protection: Integrity and Confidentiality of memory handled by SGX

• SGX allows host OS to page in/out entries from page table

• Attestation: Can verify remotely that it is running the right code
Intel SGX

- Enclave Entry and Exit
  - Some exception information leaked to the OS
    - Allows OS to handle resource management
  - Some info hidden by replacing register info with dummy info
- Dynamic Memory Allocation: Allow OS and enclave to increase memory for app
Architecture

<table>
<thead>
<tr>
<th>Component</th>
<th>Lines of Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library OS</td>
<td>millions</td>
</tr>
<tr>
<td>Shield module</td>
<td>23095</td>
</tr>
<tr>
<td>Untrusted runtime</td>
<td>7446</td>
</tr>
<tr>
<td>SGX driver</td>
<td>4520</td>
</tr>
</tbody>
</table>
Shield Module

• Primary Functionality: Protect against Iago attacks
  • Ensure that OS level functionality is meeting spec
• Reimplements Virtual Memory, Storage and Threading from the OS so that it is trusted
• Handles some other calls as well:
  • Random number generation, Windows “forks”
Evaluation

Figure 4: Performance breakdown

(a) SQL Server, TPC-E
(b) MediaWiki on Apache
Figure 5: Sensitivity to SGX instruction overhead
Limitations

• Lots of engineering effort
  • Many of the research insights were not surprising
  • Not really in the spirit of SGX
• Trusted Computing Base is HUGE
• Vulnerable to side channel attacks
  • Other systems completely broken with this
What Next?

- How do we outsource computation effectively to the cloud?
- What role does trusted hardware actually play in this space?
- How do we deal with legacy applications?